ACTION DESCRIPTIONS

ACTION 1: Dragging the cube with the mouse

Controlling the vertical and horizontal limits in the window, the TrackBall uses the function GetRotVec to get the axis and calculate the angle of rotation.

After that, we calculate a rotation matrix with the Eaa2rotMat function.

Next, we calculate the new quaternion with the GetQuatFrom2Vec function, using the initial vector and the vector who provides the GetRotVec function (which requires the mouse position).

Next we calculate the news Euler's principal axis and angle with the rotMat2Eaa using the previous calculated rotation matrix.

After that, we use the function rotM2eAngles to get the three Euler angles from the rotation matrix, and the RotVec function to get the rotation vector.

Finally, we rewrite the old rotation matrix and redraw the cube.

ACTION 2: Changing q1 values and pushing.

First, we read the new values of q1 in a new variable and calculate the Euler angles with the quat2eul function.

With that, we calculate the rotation matrix with the eAngles2rotM function.

The rest of the code does the same that the last action, calculating with the rotation matrix and Euler angles the other parameters and redraw the cube.

ACTION 3: Changing Euler axis and angle values and pushing.

In that case, we simply calculate the rotation matrix with the new values and the Eaa2rotMat function and repeat all the steps we follow in the Action 1.

ACTION 4: Changing Euler angles values and pushing.

Here, we use the eAngles2rotM function to get the rotation matrix directly and follow the same steps we followed before.

ACTION 5: Changing rotation vector values and pushing.

We extract the axis and the angle from the vector and calculate the rotation matrix with the Eaa2rotMat function, following next to that the same steps we followed in previous actions.

ACTION 6: Reset.

In this action, we simply reset all values to zero and the rotation matrix to the identity, redrawing the cube at the end.